WHAT IS CLAIMED IS:

1. In a computer system, a method for monitoring drivers, comprising,

receiving a request from a driver;

determining that the driver is to be monitored;

re-vectoring the request to a driver verifier; and

taking action in the driver verifier to monitor the driver.

2. The method of claim 1 wherein receiving a request from a driver includes receiving a function call in a kernel component of an operating system.

3. The method of claim 1 wherein determining that the driver is to be monitored includes checking a registry setting.

4. The method of claim 1 wherein the request from the driver includes a memory allocation request, and wherein taking action in the driver verifier to monitor the driver includes allocating memory space thereto from a special pool of memory.

5. The method of claim 1 wherein the request from the driver includes a memory allocation request, and wherein taking action in the driver verifier to monitor the driver includes marking memory bounding the memory space to detect improper access of the memory bounding the memory space.

6. The method of claim 1 wherein the request from the driver includes a memory deallocation request, and wherein taking action in the driver verifier to monitor the driver includes marking deallocated memory space to detect improper access of the deallocated memory space.

7. The method of claim 1 wherein taking action in the driver verifier to monitor the driver includes maintaining allocation information in at least one data structure associated with the driver.

8. The method of claim 7 wherein the request from the driver includes a memory allocation request, and wherein maintaining allocation information includes adding data corresponding to the allocation request to the data structure.

9. The method of claim 7 wherein the request from the driver includes a memory deallocation request, and wherein maintaining allocation information includes removing data corresponding to the allocation request from the data structure.

10. The method of claim 7 further comprising providing the allocation information to a user interface.

11. The method of claim 1 wherein taking action in the driver verifier to monitor the driver includes validating call parameters.

12. The method of claim 1 wherein taking action in the driver verifier to monitor the driver includes checking for resources allocated to the driver at driver unload.

13. The method of claim 1 wherein taking action in the driver verifier to monitor the driver includes simulating a low resource condition.

14. The method of claim 13 wherein simulating the low resource condition includes failing requests for memory pool allocation.

15. The method of claim 13 wherein simulating the low resource condition includes invalidating driver code and data.

16. The method of claim 1 wherein taking action in the driver verifier to monitor the driver includes checking for timers in deallocated pooled memory.

17. The method of claim 1 wherein taking action in the driver verifier includes monitoring the use of an I/O request packet by the driver.

18. The method of claim 1 wherein taking action in the driver verifier includes allocating an I/O request packet from a special pool.

19. The method of claim 1 wherein taking action in the driver verifier comprises, monitoring I/O requests traveling in the system.

20. The method of claim 1 wherein taking action in the driver verifier comprises, checking for misformed and mishandled I/O requests.

21. The method of claim 1 wherein taking action in the driver verifier comprises, modifying I/O requests to facilitate detection of mishandled I/O requests.

22. The method of claim 21 further comprising, detecting a mishandled I/O request that is a stale I/O request.

23. The method of claim 21 further comprising, detecting a mishandled I/O request that is an already completed I/O request.

24. The method of claim 21 further comprising, completing an I/O request in a restrictive environment to detect mishandling of I/O requests.

25. The method of claim 21 further comprising, detecting a mishandled I/O request that returns inconsistent results.

26. The method of claim 21 further comprising, detecting a mishandled I/O request that returns uninitialized results.

27. A computer-readable medium having computer-executable instructions, comprising:

receiving a request from a component for allocation of memory space;

determining a location of memory space to allocate;

restricting access to areas bounding the location;

allocating the memory space; and

monitoring the areas bounding the location for an access violation.

28. The computer-readable medium of claim 27 having further computer-executable instructions, comprising, detecting an access violation.

29. The computer-readable medium of claim 27 having further computer-executable instructions, comprising

receiving a request from the component for deallocation of the memory space;

restricting access to deallocated memory space; and

monitoring the deallocated memory space for an access violation.

30. The computer-readable medium of claim 29 having further computer-executable instructions, comprising, detecting an access violation in the deallocated memory space.

31. A computer-readable medium having computer-executable instructions, comprising:

receiving a plurality of requests from a component for allocation of various distinct sets memory;

allocating the memory;

tracking the space allocated to the component on each request;

receiving requests for deallocation of at least one of the sets of memory allocated to the component;

tracking the space deallocated in each received deallocation request; and

determining from the tracking whether space remains allocated to the driver at a time when the driver should have no space allocated thereto.

32. A computer-readable medium having computer-executable instructions, comprising:

receiving information corresponding to a driver unload;

determining whether resources remain associated with the driver;

and if so, generating an error.

33. The computer-readable medium of claim 32 wherein determining whether resources remain associated with the driver includes examining lists maintained by a system kernel.

34. The computer-readable medium of claim 32 wherein determining whether resources remain associated with the driver includes maintaining information tracking memory allocated to the driver and deallocated thereby.

35. A system for monitoring drivers, comprising:

an operating system component including an interface for receiving requests from drivers;

a re-vectoring component for examining the requests to determine whether requesting drivers are to be monitored; and

a driver verifier component operably connected to the re-vectoring component, the driver verifier receiving information

from the re-vectoring component for a driver that is to be monitored and executing at least one test to monitor the driver.

36. The system of claim 35 wherein the operating system component comprises a kernel component.

37. The system of claim 35 wherein the re-vectoring component determines that the driver is to be monitored based on a setting in a registry.

38. The system of claim 37 further comprising a user interface for writing driver information to the registry.

39. The system of claim 35 wherein the request from the driver includes a memory allocation request, and wherein a test by the driver verifier includes allocating memory space thereto from a special pool of memory, and marking memory bounding the memory space to detect improper access of the memory bounding the memory space.

40. The method of claim 35 wherein the request from the driver includes a memory deallocation request, and wherein a

test by the driver verifier includes deallocating the memory space, and marking the deallocated memory space to detect improper access thereof.

41. The system of claim 35 wherein a test by the driver verifier includes examining resources allocated to the driver.

42. The system of claim 41 wherein examining resources allocated to the driver includes tracking outstanding memory allocated to the driver.

43. The system of claim 41 wherein examining resources allocated to the driver includes reviewing lists maintained by the operating system component for information therein associated with the driver.

44. The method of claim 35 wherein a test performed by the driver includes validating call parameters.

45. The method of claim 35 wherein a test performed by the driver includes failing requests for memory pool allocation.

46. The method of claim 35 wherein a test performed by the driver includes invalidating driver code and data.

47. The method of claim 35 wherein a test performed by the driver includes checking for timers in deallocated pooled memory.

add b 1)